Noname manuscript No.
(will be inserted by the editor)

Peer-to-Peer Systems Design Trade-offs

A Framework exploring the balance between Blockchain and IPFS

Antonio Tenorio-Fornés · Samer Hassan · Juan Pavón

Received: date / Accepted: date

Abstract The current state of the Web, which is dominated by centralized cloud services, raises several concerns on different aspects such as governance, privacy, surveillance, and security. A way to address these issues is to decentralize the platforms by adopting new distributed technologies, such as IPFS and Blockchain, which follow a full peer-to-peer model. This work proposes a set of guidelines to design decentralized systems, taking into consideration the different trade-offs these technologies face with regard to their consistency requirements. These guidelines are then illustrated with the design of a decentralized questions and answers system. This system serves to illustrate a framework to create decentralized services and applications, that uses IPFS and Blockchain technologies and incorporates the discussion and guidelines of the paper, providing solutions for data access, data provenance and data discovery. Thus, this work proposes a framework for the design of decentralized systems and contributes a set of guidelines to decide in which cases Blockchain technology may be required, or when other technologies, such as IPFS, are sufficient.

Keywords Decentralization · Distributed Systems · P2P Systems · IPFS · Blockchain · Multi-Agent Systems

A. Tenorio-Fornés
GRASIA, Universidad Complutense de Madrid,
Decentralized Science, Decentralized Academy SL
E-mail: antoniotenorio@ucm.es

S. Hassan
Berkman Klein Center, Harvard University,
GRASIA, Universidad Complutense de Madrid
E-mail: shassan@cyber.harvard.edu

J. Pavón
Instituto de Tecnología del Conocimiento, GRASIA, Universidad Complutense de Madrid
E-mail: jpavon@fdi.ucm.es

1 Introduction

Nowadays, centralized cloud web services represent an increasingly large portion of the Internet [15]. This trend has been significantly accelerated since the emergence of the Web 2.0 model [50], in which web applications enabled user participation and user-generated contents. Thus, today's Internet activity is concentrated on highly successful web services which have dominance over their respective markets [17,24]. During recent years, there are increasing concerns on the multiple issues this situation arises, with respect to e.g. privacy [44], governance [20,17], legislation [15], surveillance [36] or security [30]. Consequently, there have been several proposals to tackle some of these issues through new legislation [8,51] or through recommendations for platform developers [26]. In parallel, these issues have triggered the emergence of a wide range of technical solutions through different forms of decentralization.

We may divide the proposed decentralized solutions in three waves. The first wave has been through "federated" technology [10,1,49], i.e. multiple central nodes communicating with each other, where users are free to choose the node to interact with. E-mail is a classic example of an open protocol which is federated, together with more recent XMPP for chatting [45], OStatus for microblogging [52], ActivityPub for social networking [54], OAuth for authentication [23], or SwellRT for real-time collaboration [40]. This approach is based on interoperability across services and servers [10,55,46]. However, many of these technologies are still hindered by several drawbacks, such as the existence of points of failure [42] and control [34], or the lack of interoperability of the data beyond a few applications [46,49].

The second wave of decentralized solutions has been achieved through fully distributed technology, i.e. P2P networks without classical servers but instead ordinary computers (different from classical cluster/grid parallel computing). There have been multiple attempts to offer P2P web services [41,31], such as Freenet for censorship-resistant communication [11], although broad adoption was mostly limited to the field of file-sharing, e.g. eDonkey, BitTorrent [13].

The third wave appears when some unresolved technical challenges with P2P solutions [33,53] became more evident. This opened the door to a new generation of solutions, most of them relying on cryptographic hashes organized in Merkle trees [37]. The advent of the first fully decentralized digital currency, Bitcoin [39], triggered a plethora of decentralized solutions based on its underlying technology, the Blockchain. In addition, another groundbreaking technology emerged around P2P storage: IPFS, or Inter-Planetary File System [3]. These two new decentralized technologies, often combined, enable a wide range of applications [18,4,19]. Furthermore, CRDTs [47] technology enabled real-time collaboration for P2P systems.

Exploring the synergies of these technologies may unveil new decentralization possibilities. IPFS is frequently used as a decentralized storage for blockchain applications. However, other non trivial combinations of these technologies may enable new decentralized systems designs.

Therefore, there is a need of frameworks and models to explore the limitations and synergies of these recent innovations. This work proposes a combination of IPFS and Blockchain technologies for the design and implementation of open distributed systems. Concretely, it presents the trade-offs decentralized technologies face, and propose design guidelines to asses the adequacy of the different considered technologies.

The rest of the paper is structured as follows. Section 2 defines characteristics of the considered distributed systems. Then, Section 3 introduces the used decentralization technologies. Section 4 discusses the trade-offs of open distributed systems design, discusses the tensions and approaches for consistency in such systems and provides design guidelines to asses whether a system may require the use of blockchain technology. Afterwards, Section 5 applies the previous sections discussions and design guidelines to propose a distributed system design, using a distributed Questions and Answer (Q&A) system as example. The conclusions follow in Section 6.

2 Open Peer to Peer Systems

The purpose of this work is to provide a framework and set of guidelines that can facilitate the design of open Peer to Peer services and applications, whose management and governance is decentralized. It focuses on open and *fully distributed peer to peer systems*. These system's characteristics are further explored in the following subsections.

2.1 Openness

Open systems should provide means for autonomous agents to enter, interact among them, and leave the system.

The concept of open system is widely applied in computing and telecommunications since a long time (see for instance standardization efforts such as the OSI model [56]). Its main idea is that services (with well specified interfaces) can be provided by different entities with their own implementation. An open system, therefore, specifies the means for communication of its entities, which can enter, interact and leave [16, 25].

The evolution of the open system is therefore highly dynamic, which makes quite complex to have a complete knowledge of the whole system state at any time. Entities only have a partial knowledge of their environment (the open system) and the only thing all of them hold in common is the ability to communicate each other [25]. In this sense, the paradigm of multi-agent systems (MAS), which assume as fundamental the autonomy and the ability to communicate of distributed entities, the *agents*, is a proper model for the development of open systems. An agent is an autonomous entity, with the assumption that its knowledge of the world is partial [22], so it tries to take the best decision (principle of rationality [48]), and interacts with other agents.

2.2 Peer to Peer Full Distribution.

Fully distributed Peer to Peer systems are composed by a network of interconnected agents that communicate and coordinate their actions without a central control entity.

Systems such as the Web and P2P File sharing programs are distributed systems composed by web servers, and computers sharing files, respectively [5, 43]. While centralized systems depend on a single component for their operation, distributed systems are resilient to the disconnection of some of their components, e.g., if a web server is disconnected, the Web will still be a functional system. However, some distributed systems still depend on single components for parts of the system to work. For instance, if a web server disconnects, their web pages will become unavailable. This work refers to *peer-to-peer systems* when referring to distributed systems that are independent from any single node.

3 Decentralization Technologies

The proposal rely on blockchain [39] and IPFS [3] decentralization technologies. This section describes these technologies and some of their underlying concepts and properties, such as content-addressability and merkle linked structures.

Content Addressability: In centralized and federated systems, content is frequently referred with addresses that include location information, the Uniform Resource Locators (URLs) [6]. However, references to content can also be independent from their location, using Universal Resource Identifiers (URIs) [27]. In peer-to-peer systems, agents cannot rely on the location of other agents for accessing content, because the content could be provided by any agent. The hash[1] of any content can be used as its URI. Thus, these hash URIs are used in multiple distributed systems such as IPFS to build scalable content-addressable networks [43, 38, 28, 3].

Merkle Links and Structures: The use of hash values (see previous subsection) to reference data in data structures was first introduced in 1987 by Merkle [37]. Complex data structures can use these links (See Figure 2 for an example). This Merkle linked structures are key to build technologies such as Git [35], Blockchain [39] and IPFS [3] among others. Section 5.2 propose the use of these structures for the data representation of the system.

Blockchain: Blockchain was the first technology that enabled a fully distributed digital currency (Bitcoin) [39], solving the double-spending problem in distributed systems. It uses a Merkle Linked list of blocks of transactions

[1] Hash functions are one-way collision-free functions, i.e. functions that, given their output, the probability to guess which input produced it is negligible.

(a Blockchain) to build a distributed ledger of transactions. To address the double spending problem, it made computationally difficult to propose a candidate for the next block in the distributed ledger and incentivized nodes to try to propose those blocks with valid transactions. Then, the protocol considers the largest observed chain the actual ledger to trust. Therefore, in order to forge a blockchain, an actor would need half of the computing power of the system, bringing security to the consistency of the data recorded in the ledger. Section 4.4 proposes the use of Blockchain to provide consistency to open distributed systems.

IPFS: Some peer-to-peer systems like P2P sharing software [43] use hash of the content to address it. Other technologies such as Git use complex Merkle-Linked Structures [35]. IPFS integrates both the use of complex Merkle-Linked structure with the data-addressability of P2P file sharing systems. The content is distributed over a peer-to-peer network. Section 5.1 proposes the use of IPFS for the storage and distribution of data in the framework.

4 Design Trade-offs of Distributed Open Systems

The design of decentralized open systems face some challenges. Unlike centralized systems, they lack a single entity deciding on the consistency of the system and a complete view of the system state. This section frames these challenges, first introducing the *CAP Theorem*, which describes the compromises between data consistency, availability, and partition resistance of distributed systems (Subsection 4.1); then presenting how the *CALM Principle* provides tools to discover if an open distributed system needs coordination technology for consistent behaviour (Subsection 4.2); next, introducing how Conflict-free Replicated Data Types (*CRDTs*) provide a solution to achieve eventual consistency for these systems without needing coordination technologies (Section 4.3), and finally explaining how *blockchain* enables such coordination while preserving decentralization when CRDTs cannot be used or the system has stronger consistency requirements (Section 4.4). This section also provides guidelines to support the design of open distributed systems.

4.1 CAP Theorem

The *CAP* Theorem [7] states that a networked data system can only hold two of these three desirable properties:

1. *C*onsistency: The requests of the distributed system behaves as if handled by a single node with updated information.
2. *A*vailability: Every request should be responded.
3. *P*artition resistance: The system is able to operate in presence of network partitions.

Given that the framework considers open systems where agents with partial information can join or leave at any moment, the *P*artition resistance is a

needed property for our proposal. Therefore, one of the most important design decisions for the systems built within the framework is to find the best balance between *C*onsistency and *A*vailability.

4.2 CALM Principle

Some queries are impossible to resolve in distributed open systems. Intuitively, in a distributed open system, since some data may not be accessible. Therefore, queries that need to take into account all the information of the system such as those counting the data that satisfy some constraints (e.g. counting the exact number of web pages that include a certain word) are impossible to resolve.

*C*onsistency *A*s *L*ogical *M*onotonicity (CALM) principle describes those queries that can be resolved in a distributed system without coordination [2]. A system is considered as *logically monotonic* if the truth of a given statement cannot change by considering new information. In such systems, the responses to distributed queries are consistent.

The designer of a distributed system can check the monotonicity of its queries as follows:

Order independence: is a needed condition for logical monotonicity [2], i.e. if the system behaviour depends on the order in which the information is received, then it is non-monotonic. For instance, in the double-spending problem, where an agent tries to spend "the same coin" twice, the state depends on which payment was done first. Therefore, it is a non-monotonic problem.

Monotonicity: By definition, if new information may revoke a previously valid response to a query, the query is non-monotonic. For instance, counting the number of positive votes for an answer in a Q&A system is non-monotonic, since new votes would change the response.

Formal analysis: can proof the logical monotonicity of a system [2].

In distributed open systems, non-monotonic queries may produce non consistent results without a coordination mechanism. Thus, in the presence of non-monotonic queries, the designer should decide on the consistency requirements of the system.

Guideline 1 *Monotonic queries can be consistently resolved in open distributed systems without coordination technologies*

Thus, in the presence of network partitions, choosing perfect consistency over availability can be implemented without coordination using logically monotonic systems. If inconsistent behaviour, like missing some votes in a Q&A system, is acceptable for the system, then coordination mechanisms are still not needed.

Guideline 2 *Consistency requirements are a design decision. If inconsistent behaviour is acceptable for the non-monotonic queries of the system, coordination technologies are not required for open distributed systems.*

Moreover, some non monotonic open distributed systems may achieve eventual consistency without coordination, as explored in next subsection.

4.3 Eventual Consistency

Eventual consistency is defined as consistency among the nodes of a distributed system once all the messages have been delivered. Conflict-free Replicated Data Types (CRDTs) proposal enable eventual consistency without coordination, such as reaching consensus or rolling back [47]. These data types can be defined on the properties of their operations. A data type is said to be a CRDT, if the possible concurrent operations are commutative.

Guideline 3 *Eventual consistency can be achieved without coordination in open distributed systems by ensuring that concurrent operations are commutative.*

Note that with eventual consistency, statements that are considered true in a given time, can become false after receiving new messages. Thus, this consistency may not be sufficient for systems with strong consistency requirements such as crypto-currencies.

CRDTs warranties eventual consistency once all the messages have been delivered. Diferent systems may tolerate different delays of these messages. For instance, while a Q&A system may ignore a vote for a long period of time, for a collaborative document, incorporating relatively old updates may be problematic, regardless the eventual consistency.

4.4 Blockchain for distributed consistency

Some non-monotonic problems, such as the double-spending problem in distributed currencies, require strong consistency. Thus, a coordination mechanism is needed to provide that consistency. Blockchain technology enabled the implementation of Bitcoin [39], the first distributed digital currency. It proposed a fully distributed coordination mechanism to establish a consensus on the order of the valid transactions. Thus, it provided consistency to a non-monotonic problem in a fully decentralized system. Indeed, blockchain can be used to provide consistency to other non monotonic systems, by establishing a consensus on the order in which the information should be considered.

Guideline 4 *The non-monotonic queries of an open distributed system with strong consistency requirements should be supported by a coordination technology such as Blockchain.*

The guidelines are summarized in Figure 1

	Weak consistency	**Eventual consistency**	**Strong consistency**
Weak availability	no need for coordination technologies (Guideline 2)		Logical Monotonicity or Blockchain (Guidelines 1, 4)
Strong availability		CRDTs (Guideline 3)	*Not possible, considering CAP Theorem*

Fig. 1 Guidelines summary

5 Designing a Distributed Question and Answers system

In this section, the trade-offs and design guidelines introduced in this paper are presented through a running example of a simple Q&A system, such as the well-known Stack Overflow[2]. The balance between availability and consistency in the system is discussed, and the need for blockchain technology is assesed.

The proposed system architecture relies on IPFS for fully distributed data storage, public-key identities for data provenance, and a peer-to-peer network for communication. This section introduces how data access, data provenance and data discovery are provided by the proposal.

5.1 Accessing data

In centralized Q&A systems such as Stack Overflow the data is addressed and accessed using a location-centric model. i.e. a server is responsible to provide the data. For instance, a user may search for responses to a programming problem on Stack Overflow website.

The use of content-addressable models for data access provide a fully distributed alternative. Our architecture relies on the IPFS newtwork to distribute the data as Merkle-linked structures. This data structures provide both a Merkle-linked structure and data-addressability [3]. Concretelly, the data in the system is composed by key-value records and by named directed Merkle-links to other data (as depicted in Figure 2). This data may be provided by any agent of the system.

5.2 Data provenance

In centralized and federated systems, the trustworthiness of the data is provided through direct connection to trusted servers, e.g. the user of a centralized Q&A system trust a server for not hiding or altering the information of the system. Fully decentralized alternatives can also be considered to obtain trustworthy data.

We propose the use of asymmetric cryptography identities to provide trustworthy provenance of data. Data that is digitally signed by trusted identities is

[2] https://stackoverflow.com

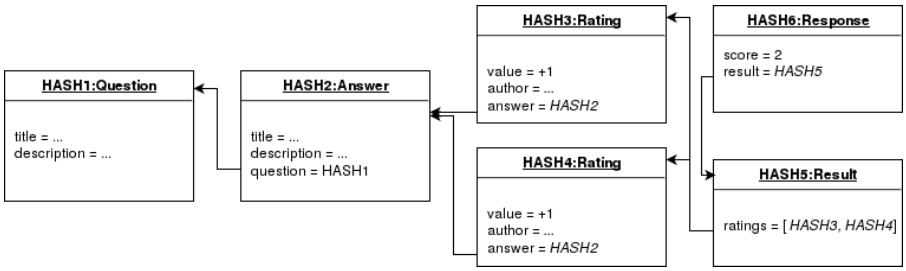

Fig. 2 Merkle linked data of an example Q&A system (such as Stack Overflow)

trusted in the system. Following the technological choices of the architecture, the use of IPNS [3] or Ethereum [9] identity infrastructure can be used.

Following our Q&A system example, every question, answer and vote is digitally signed by the authors. Replicating the behaviour of Stack Overflow, every user can submit questions and answers to the system. Thus, every signed question or answer is consider valid. A simple version of the system may consider every question, answer and vote valid, thus having weak consistency requirements. Such system would not need coordination technologies (Guideline 2) to work. However, systems such as Stack Overflow implement strategies to avoid system abuses, for instance, it only allows to vote to authors with at least 15 reputation points. Five reputation points are earned with each positive vote to a question or answer. Thus, to implement such strategies, our system should only allow the votes of users with at least 3 positive votes. Since this votes also have to be valid, the vote verification is recursive, until it reaches a trusted base case, e.g. identities that were initially allowed to vote without reputation in the system.

If negative votes are not considered in the system, answering whether a vote is valid is a non-monotonic problem. Thus, it can be implemented in a distributed system with strong consistency without coordination mechanisms (Guideline 1). However, the recursive nature of the example shows how the size and complexity of the data needed to trust a response may not be trivial.

The consideration of negative votes to questions and answers that would decrease the reputation of the authors adds complexity to the problem. The question of whether an identity has at least 15 reputation points is no longer monotonic, since observing new negative votes may change the results. Fortunately, adding and subtracting values to a number are commutative operations. Thus, and following CRDTs proposal, we could chose availability over consistency, and be able to operate in the system while not knowing all the up and down votes while trusting that eventual consistency will be achieved (Guideline 3).

Furthermore, digital signatures may not be enough to prove the authorship in the system. A malicious agent may sign data previously authored by other agents. Deciding then which was the first author become a non-monotonic problem that cannot be resolved with strong consistency without coordination.

This problem is alike the double-spending problem, and could be resolved using blockchain[3] if the designer considers that the system requires such strong consistency (Guideline 4).

Non-monotonic searches (see Section 4.2) with strong consistency requirements, such as getting exact number of votes of a question, may need the use of a blockchain as coordination mechanism. For instance, the votes of a Q&A system or the authorship of questions and answers could be registered in a blockchain to provide consistency to those queries. Our architecture proposes the development of smart contracts using Ethereum [9] to provide such consistency for these systems.

5.3 Data Discovery using a Trustless Distributed Protocol

To discover data in our open and distributed system we propose the use of a query protocol. The queries of the system state the constraint that the responses must satisfy. For instance, a question that contains a given text can be searched in a Q&A system. The query can also constraint the structure of the response (e.g. it has more than one answer and more than one positive vote).

Additionally, a score function can be defined to sort the valid responses. For instance, the the questions containing some text can be ranked by the number of positive votes.

Following, the protocol interactions (Figure 3) are described:

1. An agent sends a query (with constraints and a score function).
2. Any agent can reply, with a response consisting of a content-centric link to the data satisfying the query and its corresponding score.
3. The querying agent access the data, and verifies the responses and scores.

This protocol presents the following characteristics:

1. Lightweight communication: Responses consist of a short link and a numeric value. Their length is then a few bytes long while they may represent complex large data structures.
2. Early distributed ranking: Responses may be ranked without accessing their data.
3. Trustless ranking and validity: The validity and ranking of the responses can be assessed without trusting the agents providing the responses or the data.

The protocol can be implemented using: 1) Merkle-linked data distributed over IPFS. 2) Javascript pure functions to express query constraints and score functions, using the JavaScript implementation of IPFS, and 3) A bus model for distributed systems communication [29] over IPFS pub-sub channels. Thus, it would enable the implementation of distributed open systems without with

[3] Registering first a pre-commitment that links to the author's identity to the data, to avoid malicious agents to claim the authorship of the data as soon as they see it.

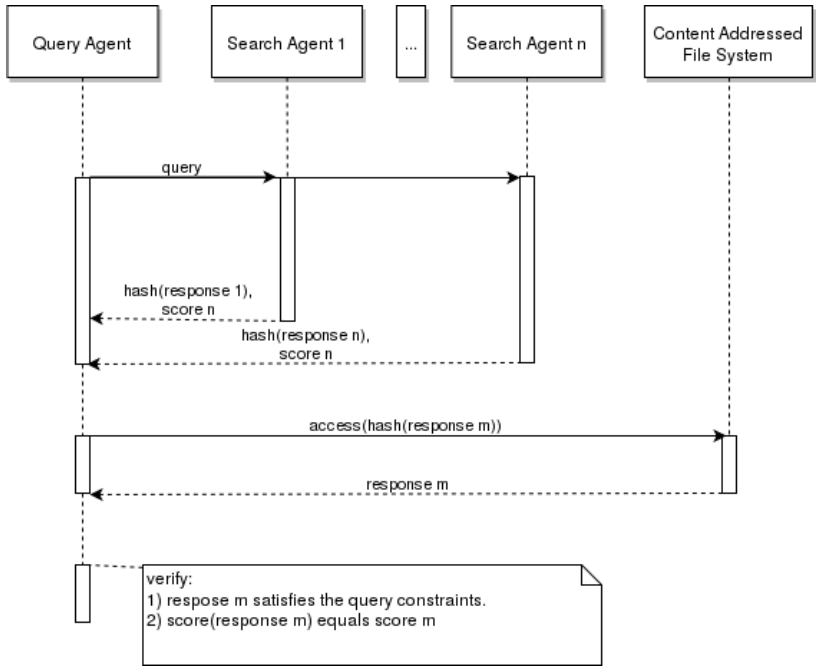

Fig. 3 Distributed Discovery Protocol UML Sequence Diagram

different consistency and availability requirements (See Figure 1 for a summary of the guidelines for those different requirements).

6 Discussion and conclusions

This work introduces the tensions between consistency, availability and partition resistance in current fully distributed systems. It explores the possibilities and limitations of different approaches and technologies, providing guidelines to design these fully distributed systems. The guidelines help to assess whether blockchain technology may be needed for a distributed system. Four guidelines provide alternatives depending on the consistency and availability requirements of the system. The paper claims that these consistency and availability requirements are design decisions, and that some systems may not have strong requirements for either of them, thus not needing advanced technologies to enhance the coordination or availability (Guideline 2). For solutions that require strong consistency, logical monotonic systems can provide such consistency without coordination (Guideline 1). However, not all problems are non-monotonic, and in that case a blockchain is required to provide such consistency and maintain the system decentralization (Guildeline 4). For systems with weaker consistency requirements, CRDTs offer an alternative that favor high availability while relaxing their consistency requirements to eventual consistency (Guideline 3).

The paper then presents an architecture, which is illustrated with a running example of a Q&A system. In this proposal, the data is represented as Merkle-linked structures and distributed with IPFS. Asymmetric cryptography provides trust to the data provenance of the distributed system. Ethereum technology is proposed as the blockchain-based coordination framework to support the non-monotonic strong consistency requirements these systems may have. A query communication protocol enables the data discovery in the open distributed system, providing ranked responses and trust-less verification of responses.

This proposal faces some limitations and challenges, as other blockchain-based and distributed technologies, such as privacy [21,14] and sustainability [12]. Furthermore, the design of distributed systems following our proposal should consider security concerns of distributed systems such as *sybil attacks* [39] and *generation attacks* [32]. Still, the sustainability, and privacy of decentralized technologies is often better than the centralized alternatives [55].

Future work would help to consolidate and validate the contributions of this paper. Studying the efficiency and performance of the system, the proposal and implementation of new applications, the identification of more suitable network topologies and protocols, or the use of specialized agents such as search agents for specific applications, are some of the opportunities to explore.

Decentralization technologies offer an opportunity to solve some of the challenges of the current Internet. This paper has introduced design guidelines a framework to design and build these systems using the potentials of new decentralizing technologies.

7 Acknowledgments

This work was partially supported by the project P2P Models (`https://p2pmodels.eu`) funded by the European Research Council ERC-2017-STG (grant no.: 759207), Decentralized Science (`https://decentralized.science`) funded by European Union's Horizon 2020 research and innovation programme within the framework of the LEDGER Project (grant agreement No82526). and ColoSAAL (`http://grasia.fdi.ucm.es/colosaal/`), funded by the Spanish Ministry of Economy and Competitiveness (TIN2014-57028-R).

References

1. Alsaleh, M., Adams, C.: Enhancing consumer privacy in the liberty alliance identity federation and web services frameworks. In: International Workshop on Privacy Enhancing Technologies, pp. 59–77. Springer, Berlin, Heidelberg (2006)
2. Alvaro, P., Conway, N., Hellerstein, J.M., Marczak, W.R.: Consistency analysis in bloom: a calm and collected approach. In: CIDR 2011 - 5th Biennial Conference on Innovative Data Systems Research, Conference Proceedings, pp. 249–260. Asilomar, California (2011)
3. Benet, J.: Ipfs-content addressed, versioned, p2p file system. arXiv preprint arXiv:1407.3561 (2014)

4. Benshoof, B., Rosen, A., Bourgeois, A.G., Harrison, R.W.: Distributed decentralized domain name service. In: Parallel and Distributed Processing Symposium Workshops, 2016 IEEE International, pp. 1279–1287. IEEE, Chicago, IL, USA (2016)
5. Berners-Lee, T.: Long live the web. Scientific American **303**(6), 80–85 (2010)
6. Berners-Lee, T., Masinter, L., McCahill, M.: Uniform resource locators (url). Tech. rep. (1994)
7. Brewer, E.: Cap twelve years later: How the" rules" have changed. Computer **45**(2), 23–29 (2012)
8. Brown, I., Marsden, C.T.: Regulating code: Good governance and better regulation in the information age. MIT Press (2013)
9. Buterin, V.: Ethereum: A next-generation smart contract and decentralized application platform. URL https://github. com/ethereum/wiki/wiki/% 5BEnglish% 5D-White-Paper (2014)
10. Cabello, F., Franco, M.G., Haché, A.: The social web beyond'walled gardens': Interoperability, federation and the case of lorea/n-1. PsychNology Journal **11**(1), 43–65 (2013)
11. Clarke, I., Sandberg, O., Wiley, B., Hong, T.W.: Freenet: A distributed anonymous information storage and retrieval system. In: Designing privacy enhancing technologies, pp. 46–66. Springer (2001)
12. Cohen, B.: Incentives build robustness in bittorrent. In: Workshop on Economics of Peer-to-Peer systems, vol. 6, pp. 68–72 (2003)
13. Cohen, B.: The bittorrent protocol specification (2008). URL `http://www.bittorrent.org/beps/bep_0003.html`
14. De Filippi, P.: The interplay between decentralization and privacy: the case of blockchain technologies (2016)
15. De Filippi, P., McCarthy, S.: Cloud computing: Centralization and data sovereignty. European Journal of Law and Technology **3**(2) (2012). URL `https://ssrn.com/abstract=2167372`
16. Demazeau, Y., Costa, A.R.: Populations and publishers in open multi-agent systems. In: Proceedings of the 1st National Symposium on Parallel and Distributed AI, pp. 1–13 (1996)
17. Dolata, U.: Apple, Amazon, Google, Facebook, Microsoft: Market concentration-competition-innovation strategies. Tech. rep., Stuttgarter Beiträge zur Organisations- und Innovationsforschung, SOI Discussion Paper (2017)
18. Faísca, J.G., Rogado, J.Q.: Decentralized semantic identity. In: Proceedings of the 12th International Conference on Semantic Systems, SEMANTiCS 2016, pp. 177–180. ACM, New York, NY, USA (2016). DOI 10.1145/2993318.2993348. URL `http://doi.acm.org/10.1145/2993318.2993348`
19. Faisca, J.G., Rogado, J.Q.: Personal cloud interoperability. In: 2016 IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM), pp. 1–3 (2016). DOI 10.1109/WoWMoM.2016.7523546
20. Fuster Morell, M.: Governance of online creation communities: Provision of infrastructure for the building of digital commons. Ph.D. thesis (2010)
21. Greschbach, B., Kreitz, G., Buchegger, S.: The devil is in the metadata—new privacy challenges in decentralised online social networks. In: 2012 IEEE International Conference on Pervasive Computing and Communications Workshops, pp. 333–339 (2012). DOI 10.1109/PerComW.2012.6197506
22. Groen, F.C., Spaan, M.T., Kok, J.R., Pavlin, G.: Real world multi-agent systems: information sharing, coordination and planning. In: International Tbilisi Symposium on Logic, Language, and Computation, pp. 154–165. Springer (2005)
23. Hammer-Lahav, E.: The OAuth 1.0 protocol, internet engineering task force (IETF) (2010). URL `https://tools.ietf.org/html/rfc5849`
24. Haucap, J., Heimeshoff, U.: Google, Facebook, Amazon, eBay: Is the Internet driving competition or market monopolization? International Economics and Economic Policy **11**(1-2), 49–61 (2014)
25. Hewitt, C., De Jong, P.: Open systems. In: On Conceptual Modelling, pp. 147–164. Springer (1984)

26. Hill, B.M.: Franklin street statement on freedom and network services. Autonomo. us (2008). URL https://web-beta.archive.org/web/20151006113744/http://autonomo.us/2008/07/14/franklin-street-statement/
27. Jacobs, I.: Uris, addressability, and the use of http get and post. World Wide Web Consortium, TAG Finding **13**, 28 (2004)
28. Janak, J., Lee, J.W., Schulzrinne, H.: Grand: Git revisions as named data (2011)
29. Jun, K., Boloni, L., Palacz, K., Marinescu, D.C.: Agent-based resource discovery. In: Heterogeneous Computing Workshop, 2000.(HCW 2000) Proceedings. 9th, pp. 43–52. IEEE (2000)
30. Kandukuri, B.R., Rakshit, A., et al.: Cloud security issues. In: Services Computing, 2009. SCC'09. IEEE International Conference on, pp. 517–520. IEEE (2009)
31. Kermarrec, A.M.: Towards a personalized internet: a case for a full decentralization. Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences **371**(1987), 20120,380 (2013)
32. Labs, P.: Filecoin: A decentralized storage network (2017). URL https://filecoin.io/filecoin.pdf
33. Lamport, L., Shostak, R., Pease, M.: The byzantine generals problem. ACM Transactions on Programming Languages and Systems (TOPLAS) **4**(3), 382–401 (1982)
34. Levison, L.: Secrets, lies and snowden's email: why i was forced to shut down lavabit. The Guardian (2014)
35. Loeliger, J., McCullough, M.: Version Control with Git: Powerful tools and techniques for collaborative software development. " O'Reilly Media, Inc." (2012)
36. Lyon, D.: Surveillance, snowden, and big data: Capacities, consequences, critique. Big Data & Society **1**(2), 2053951714541,861 (2014)
37. Merkle, R.C.: A digital signature based on a conventional encryption function. In: Conference on the Theory and Application of Cryptographic Techniques, pp. 369–378. Springer (1987)
38. Morris, R., Kaashoek, M.F., Karger, D., Balakrishnan, H., Stoica, I., Liben-Nowell, D., Dabek, F.: Chord: A scalable peer-to-peer look-up protocol for internet applications. IEEE/ACM Transactions On Networking **11**(1), 17–32 (2003)
39. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (2008)
40. Ojanguren-Menendez, P., Tenorio-Fornés, A., Hassan, S.: Building real-time collaborative applications with a federated architecture. IJIMAI **3**(5), 47–52 (2015)
41. Pouwelse, J.A., Garbacki, P., Wang, J., Bakker, A., Yang, J., Iosup, A., Epema, D.H., Reinders, M., Van Steen, M.R., Sips, H.J.: Tribler: a social-based peer-to-peer system. Concurrency and computation: Practice and experience **20**(2), 127–138 (2008)
42. Rajabi, E., Sanchez-Alonso, S., Sicilia, M.A.: Analyzing broken links on the web of data: an experiment with dbpedia. Journal of the Association for Information Science and Technology **65**(8), 1721–1727 (2014)
43. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Shenker, S.: A scalable content-addressable network, vol. 31. ACM (2001)
44. Rubinstein, I., Van Hoboken, J.: Privacy and security in the cloud: some realism about technical solutions to transnational surveillance in the post-snowden era (2014)
45. Saint-Andre, P.: Extensible messaging and presence protocol (xmpp): Address format. Tech. rep. (2015)
46. Sambra, A.V., Mansour, E., Hawke, S., Zereba, M., Greco, N., Ghanem, A., Zagidulin, D., Aboulnaga, A., Berners-Lee, T.: Solid: A platform for decentralized social applications based on linked data Technical report
47. Shapiro, M., Preguiça, N., Baquero, C., Zawirski, M.: Conflict-free replicated data types. In: Symposium on Self-Stabilizing Systems, pp. 386–400. Springer (2011)
48. Simon, H.A.: Cognitive science: The newest science of the artificial. Cognitive science **4**(1), 33–46 (1980)
49. Tapiador, A., Hassan, S.: Understanding federation: An analytical framework for the interoperability of social networking sites. arXiv preprint arXiv:1805.06474 (2018)
50. Tim, O.: What is web 2.0? design patterns and business models for the next generation of software (2005)
51. Voss, W.G.: European union data privacy law reform: General data protection regulation, privacy shield, and the right to delisting (2017)

52. W3C: The basics - OStatus community group. http://www.w3.org/community/ostatus/wiki/The_Basics (2012). Accessed: 2016-10-11
53. Wachs, M., Schanzenbach, M., Grothoff, C.: On the feasibility of a censorship resistant decentralized name system. In: Foundations and Practice of Security, pp. 19–30. Springer (2014)
54. Webber Lemmer, C., Tallon, J., Shepherd, E., Guy, A., Prodromou, E.: ActivityPub (2018). URL https://www.w3.org/TR/activitypub/
55. Yeung, C.m.A., Liccardi, I., Lu, K., Seneviratne, O., Berners-Lee, T.: Decentralization: The future of online social networking. In: W3C Workshop on the Future of Social Networking Position Papers, vol. 2, pp. 2–7 (2009)
56. Zimmermann, H.: Osi reference model–the iso model of architecture for open systems interconnection. IEEE Transactions on communications **28**(4), 425–432 (1980)